

CSS BOX MODEL

```
.css {  
   model  
}
```



SoftUni Team
Technical Trainers



SoftUni

Software University
<https://softuni.bg>

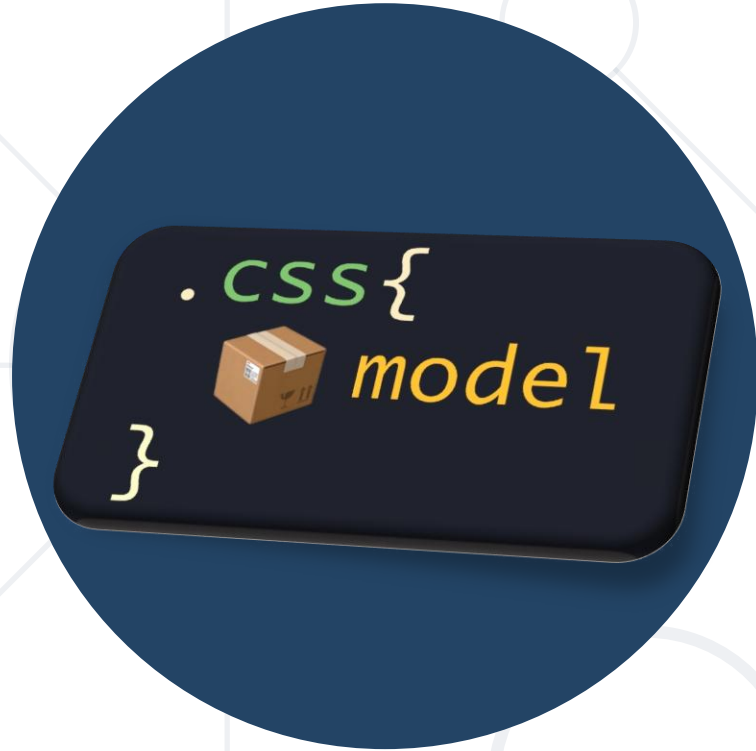
Table of Contents

1. CSS Box Model
2. Block and Inline Elements
3. Width and Height
4. Padding, Margin and Border
5. Box Sizing



sli.do

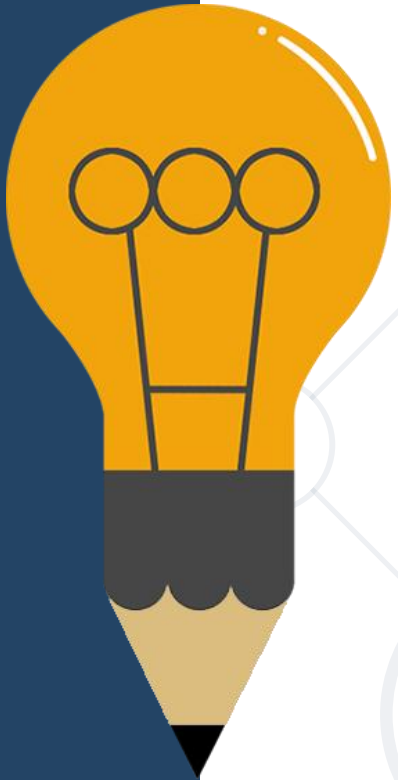
#HTML-CSS



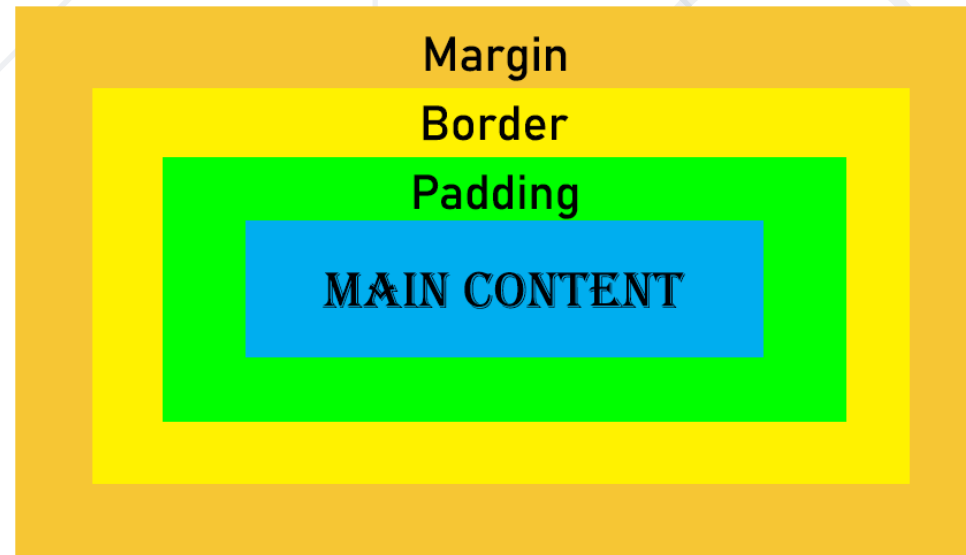
CSS Box Model

What is CSS Box Model?

- The CSS box model is essentially a **box** that wraps around every HTML element
- All HTML elements can be considered as boxes
- The term "**box model**" is used when talking about design and layout
- CSS box model consists of: margins, borders, padding, and the actual content



- Content Box – the area where your content is displayed, which can be sized using properties like width and height
- Padding Box – the padding sits around the content
- Border Box – the border box wraps the content and any padding
- Margin Box – the margin wrapping the content, padding and border





Block and Inline Elements

- HTML is made up of various elements that act as the **building blocks** of web pages
- CSS has two different types of boxes — **block** and **inline**
 - Block Elements
 - Inline Elements
 - Inline-block Elements

- Block element: starts on a **new line**, and fills up the horizontal space left and right on the web page
- Some examples of block elements are:
 - main, header, article, section, fieldset, nav, ul, ol, li, form, h1-h6, p, div
- You can add **margins** and **padding** on **all four sides** of any block element

Block Elements - Example

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width" />
    <title></title>
  </head>
  <body>
    <div>This is my div tag.</div>
    <p>This is my paragraph tag.</p>
  </body>
</html>
```

This is my div tag.

This is my paragraph tag.

- Inline element: **don't start** on a new line. They appear on the same line as the content and tags beside them
- Some examples of inline-block elements are:
 - a, label, map, span, strong, em, i, img, textarea, input, button, select
- You can add margins and padding just on **right** and **left** sides of any inline element

Inline Elements - Example

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width" />
    <title></title>
  </head>
  <body>
    <div>This is my <strong>div</strong> tag.</div>
    <p>This is my <span>paragraph</span> tag.</p>
  </body>
</html>
```

This is my **div** tag.

This is my **paragraph** tag.

- Inline-block elements are similar to inline elements
- They can have padding and margins added on **all four sides**
- You have to declare **display: inline-block** in your CSS code
- One common use for using inline-block is for creating navigation links horizontally

Inline-Block Elements - Example

```
<ul>
  <li>Home</li>
  <li>About Us</li>
  <li>Clients</li>
  <li>Contacts</li>
</ul>
```

```
ul {
  background-color: #F0B27A;
  padding: 20px;
  list-style-type: none;
  text-align: center;
}

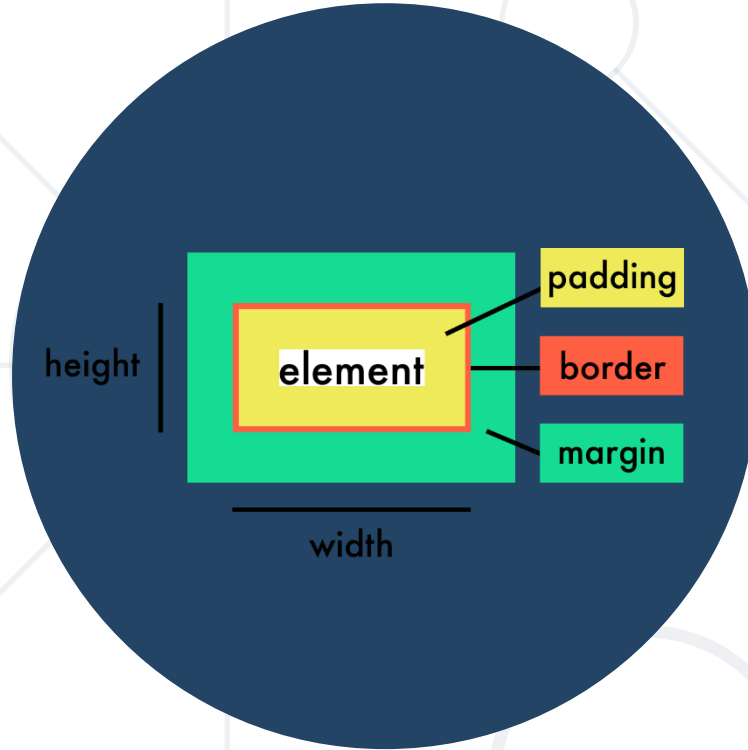
li {
  display: inline-block;
  padding: 0 20px;
  font-size: 20px;
}
```

Home

About Us

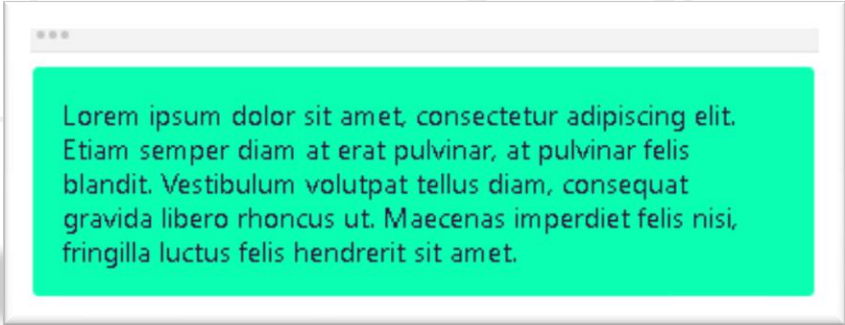
Clients

Contact

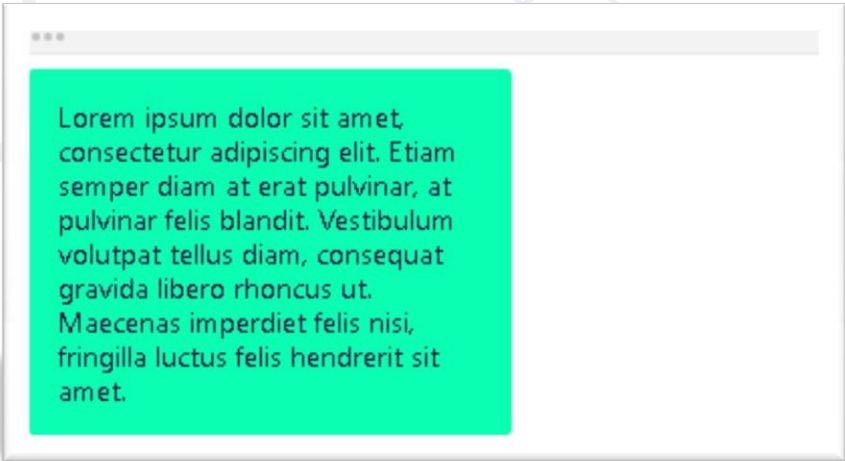


Width and Height

- Width – defines the width of the element
 - **width: auto;** - the element will **automatically** adjust its width to allow its content to be displayed correctly
 - **width: 240px;** - you can use numeric values like **pixels**, **(r)em**, **percentages...**

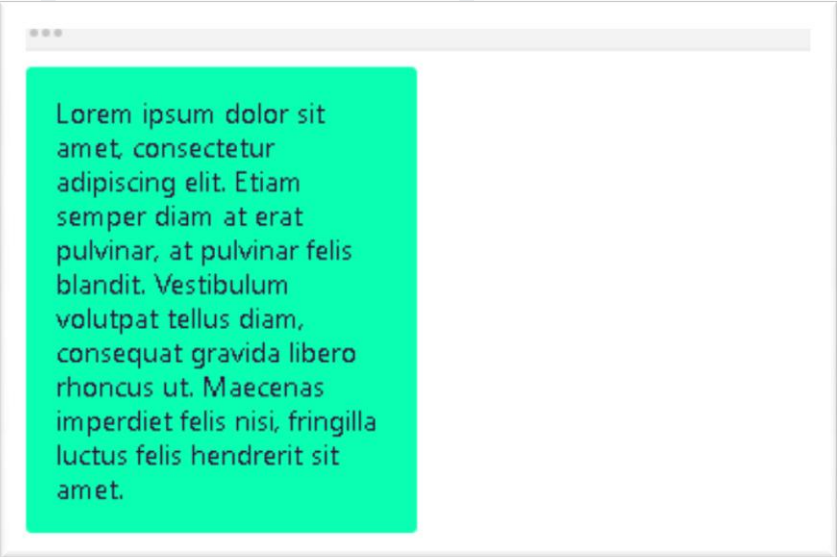


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet.



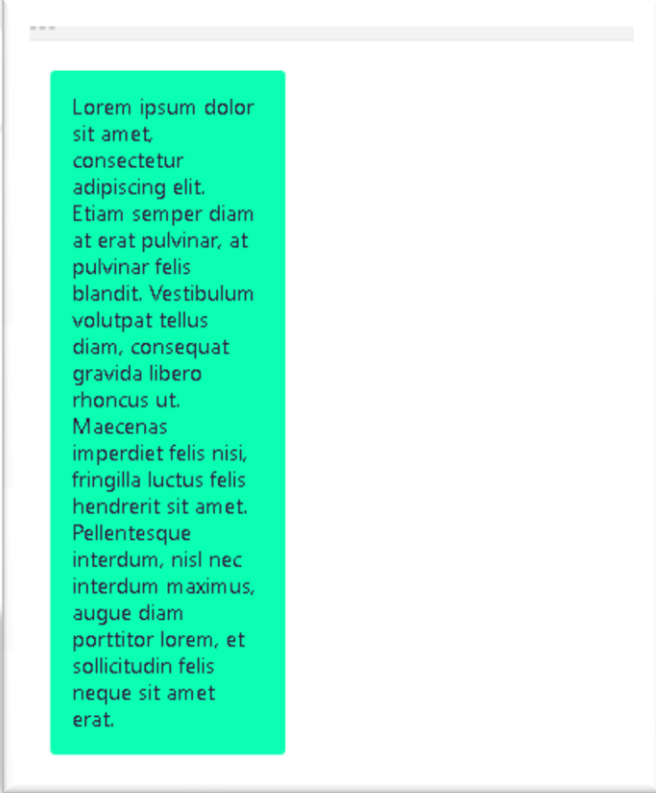
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet.

- **width: 50%;** - if you use **percentages**, the value is relative to the container's width

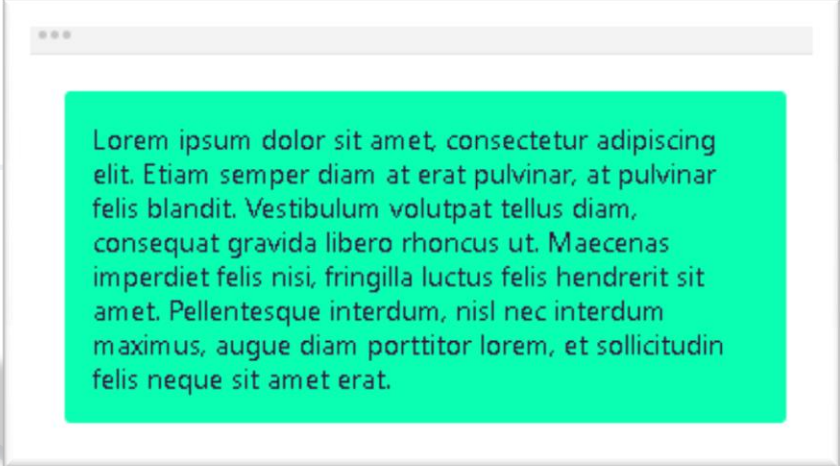


Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Etiam
semper diam at erat
pulvinar, at pulvinar felis
blandit. Vestibulum
volutpat tellus diam,
consequat gravida libero
rhoncus ut. Maecenas
imperdiet felis nisi, fringilla
luctus felis hendrerit sit
amet.

- Max-width – defines the **maximum** width the element can be
 - **max-width: none;** - the element has **no limit** in terms of width
 - **max-width: 150px;**

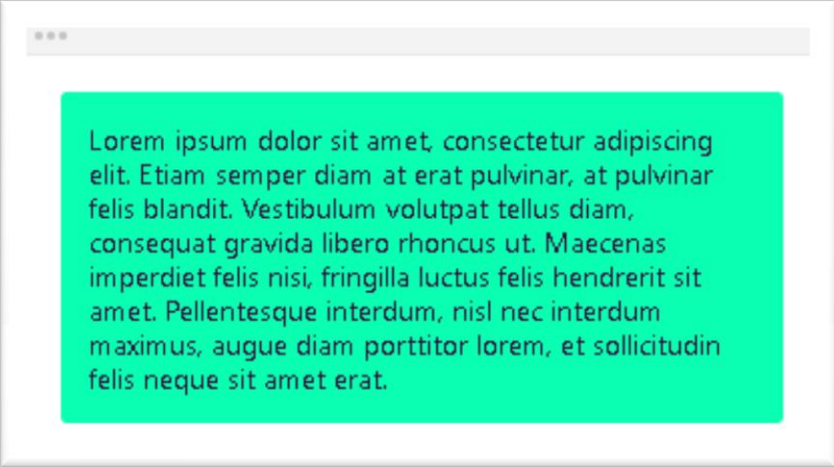


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum maximus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.



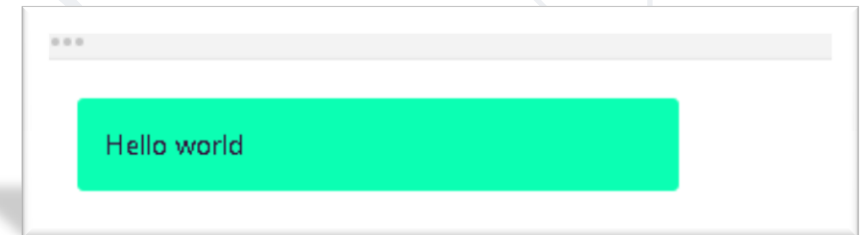
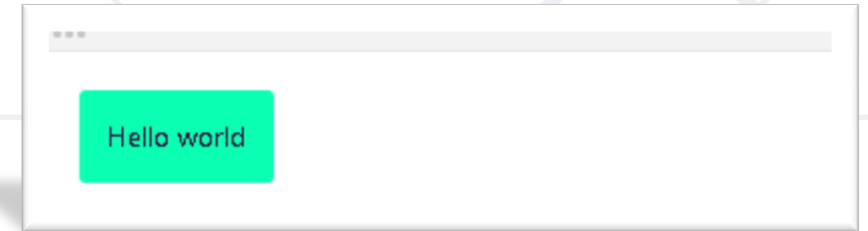
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum maximus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.

- **max-width: 2000px;** - you can use numeric values like **pixels**, **(r)em**, **percentages...**
- If the *maximum* width is **larger** than the element's *actual* width, the max width has **no effect**



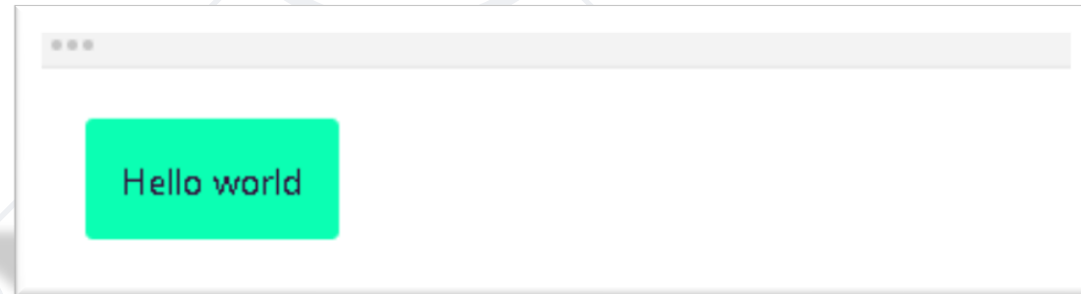
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum maximus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.

- Min-width – defines the **minimum** width the element
 - **min-width: 0;** - the element has **no minimum** width
 - **min-width: 300px;** - if the *minimum* width is **larger** than the element's *actual* width, the min width will be applied

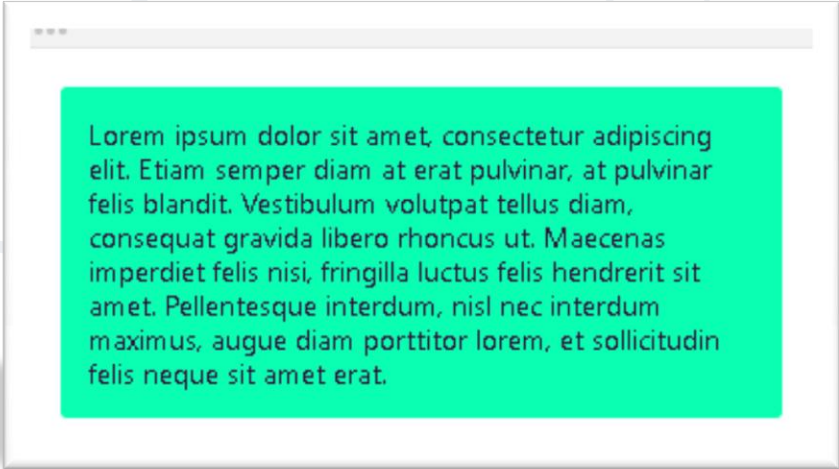


- **min-width: 5px;**

If the *minimum* width is **smaller** than the element's *actual* width, the min width has **no effect**

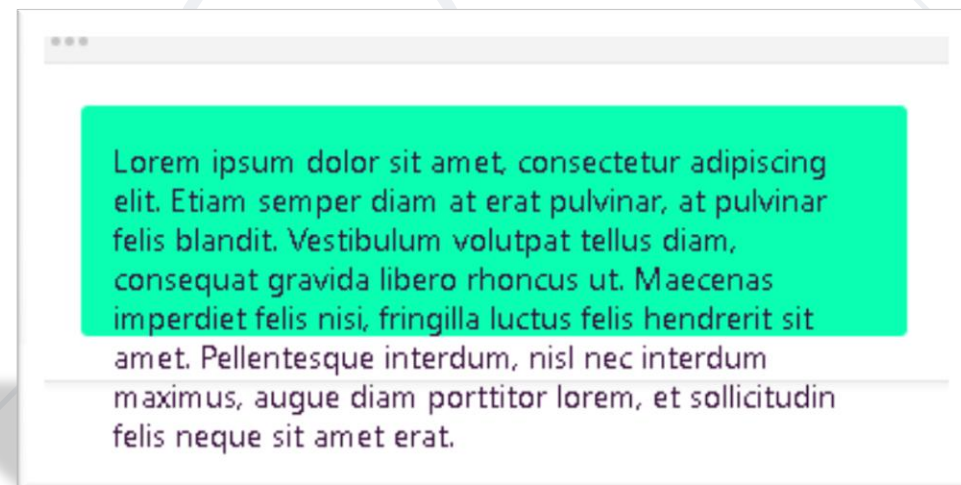


- Height – defines the height of the element
 - **height: auto;** - the element will **automatically** adjust its height to allow its content to be displayed correctly

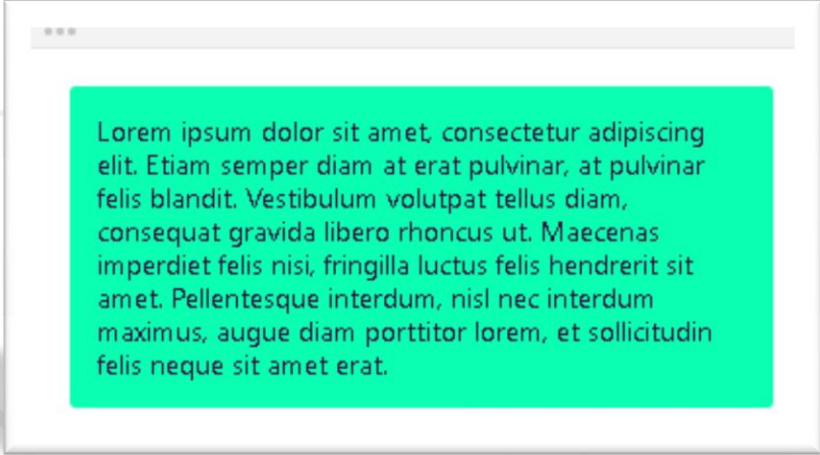


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum maximus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.

- **height: 100px;** - you can use numeric values like **pixels**, **(r)em**, **percentages...**
- If the content does not fit within the specified height, it will **overflow**
- How the container will handle this overflowing content is defined by the **overflow property**

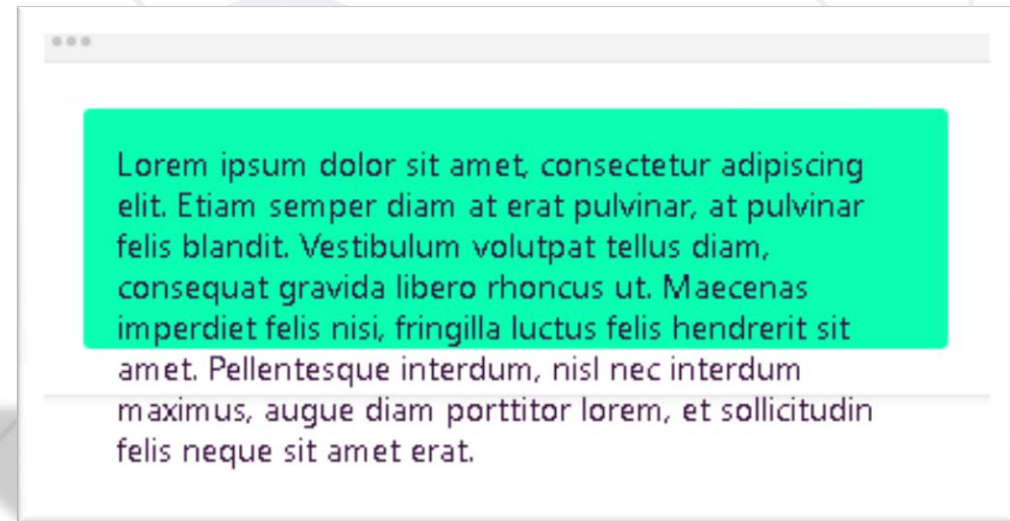


- Max-height – defines the maximum height the element can be
 - **max-height: none;** - the element has **no limit** in terms of height
 - **max-height: 2000px;** - if the *maximum* height is **larger** than the element's *actual* height, the max height has **no effect**



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum maximus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.

- **max-height: 100px;**
- If the content does not fit within the maximum height, it will **overflow**
- How the container will handle this overflowing content is defined by the **overflow property**



- Min-height – defines the minimum height the element
 - **min-height: 0;** - the element has **no minimum** height

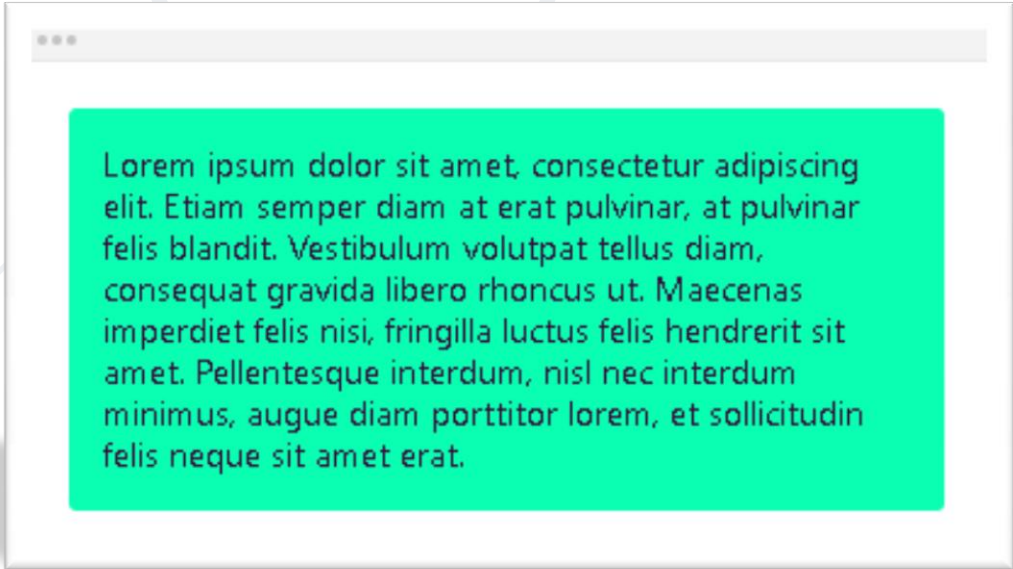
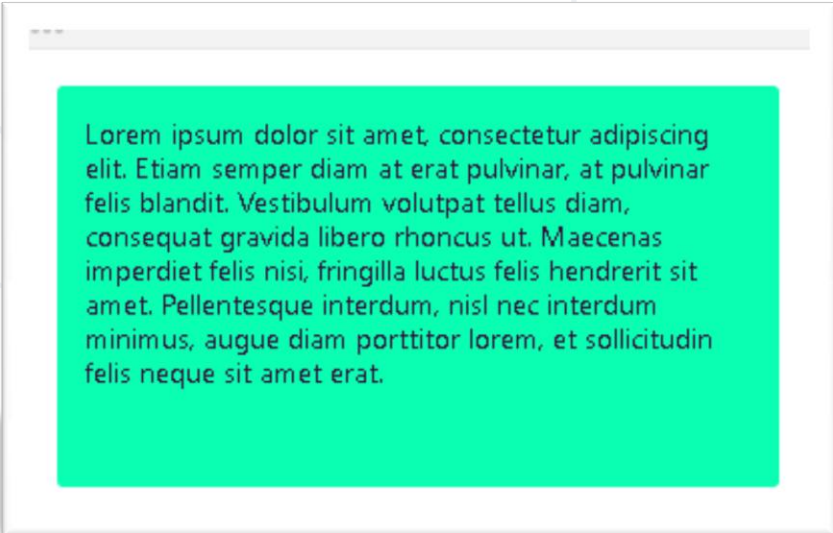


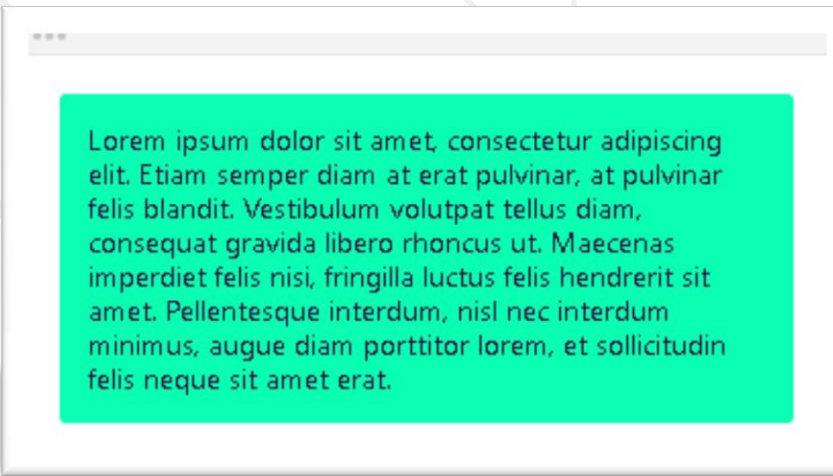
Diagram illustrating a browser window with a content area that has a minimum height of 0. The content area is highlighted in red, and the text inside is grayed out, indicating it is not visible due to the min-height constraint.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum minimus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.

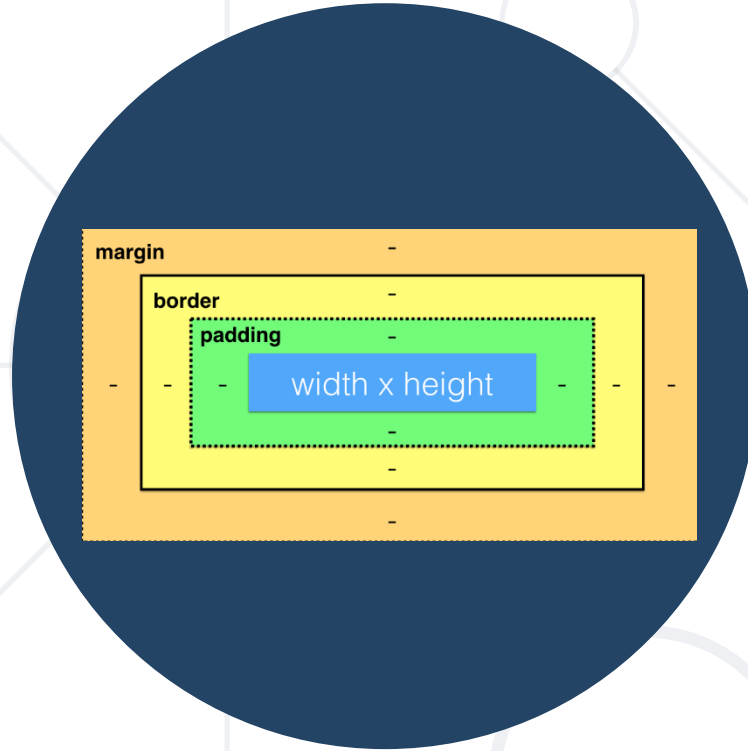
- **min-height: 200px;** - if the *minimum* height is **larger** than the element's *actual* height, the min height will be applied
- **min-height: 5px;** - if the *minimum* height is **smaller** than the element's *actual* height, the min height has **no effect**



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum minimus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.

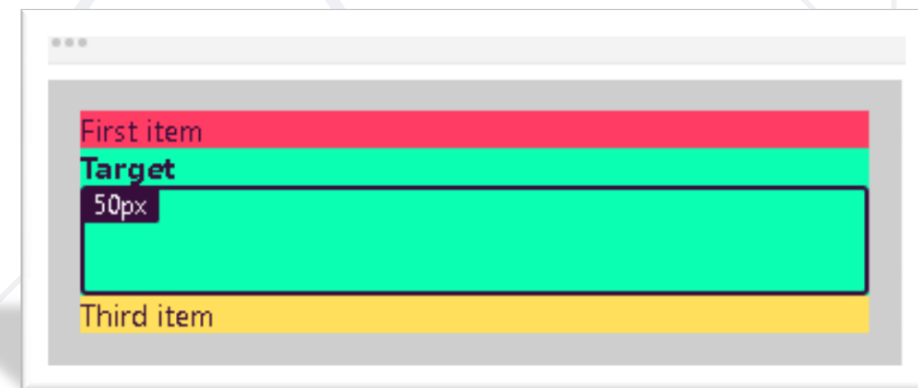
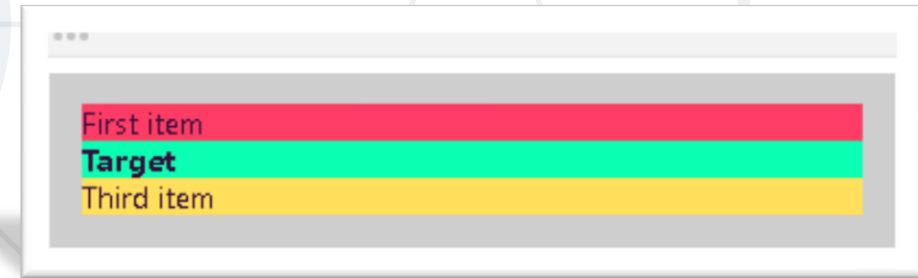


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Pellentesque interdum, nisl nec interdum minimus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat.

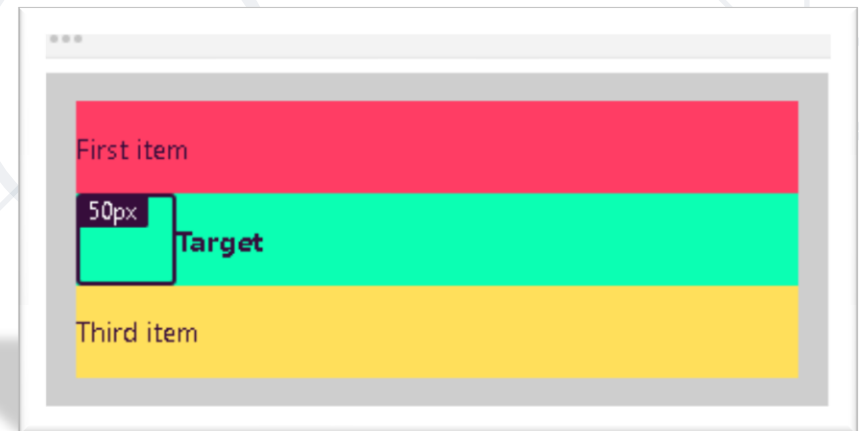


Margin, Padding and Border

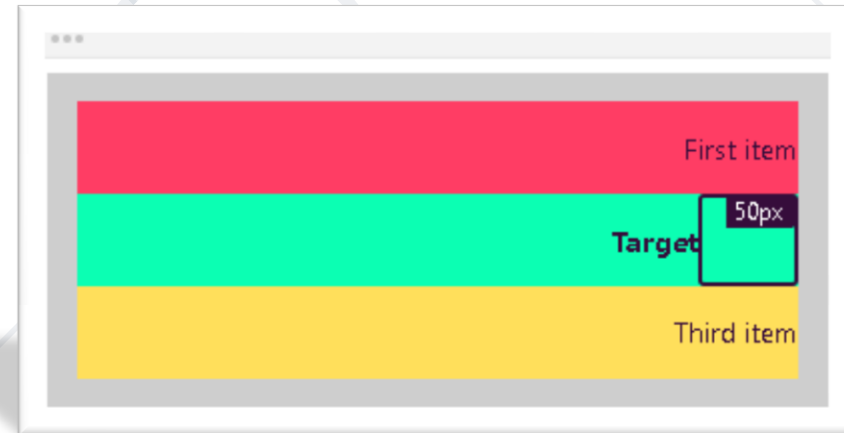
- **Padding** – defines the space **inside** the element
- Padding-bottom:
 - padding-bottom: 0;
 - padding-bottom: 50px;



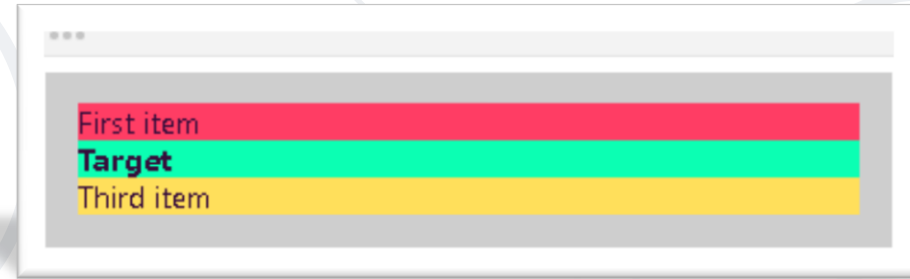
- `padding-left:`
 - `padding-left: 0;`
 - `padding-left: 50px;`



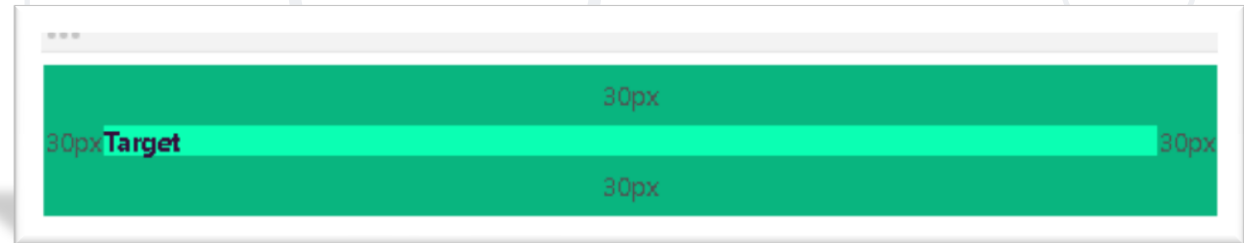
- **Padding-right:**
 - `padding-right: 0;`
 - `padding-right: 50px;`



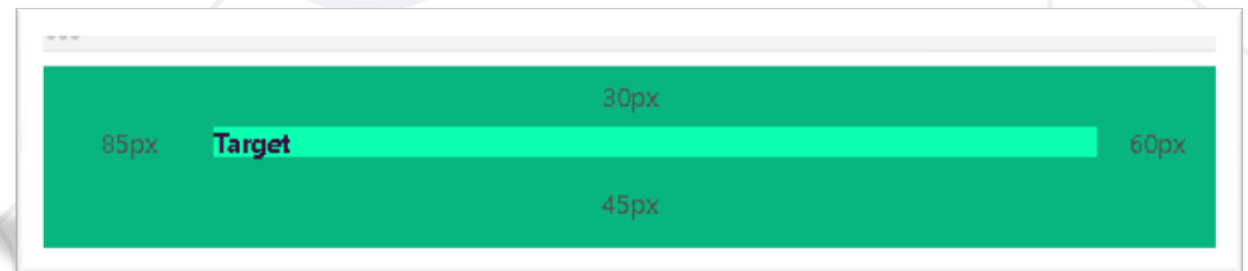
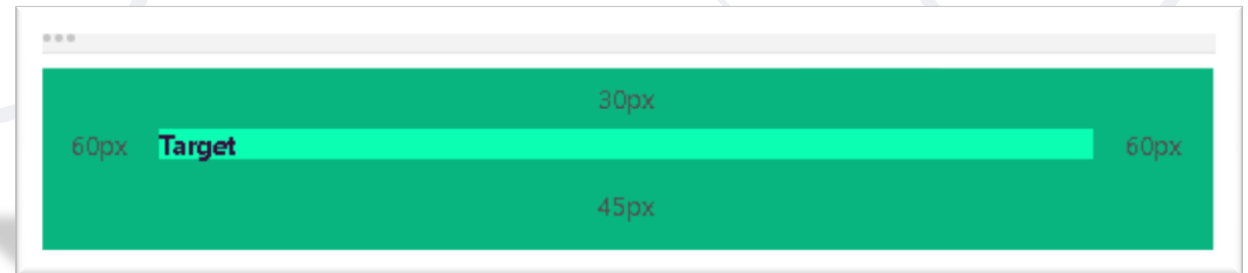
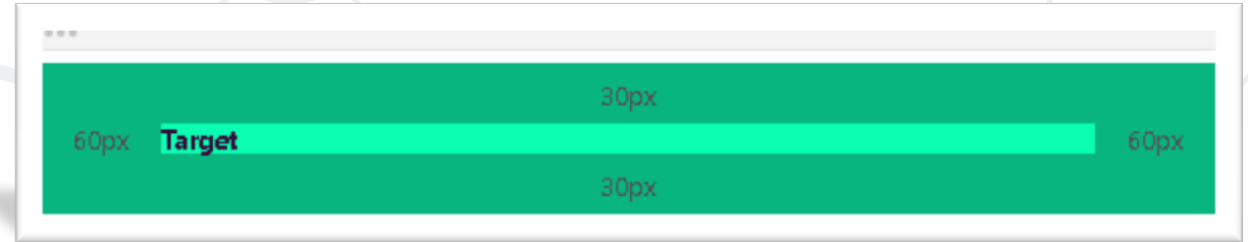
- **Padding-top:**
 - `padding-top: 0;`
 - `padding-top: 50px;`



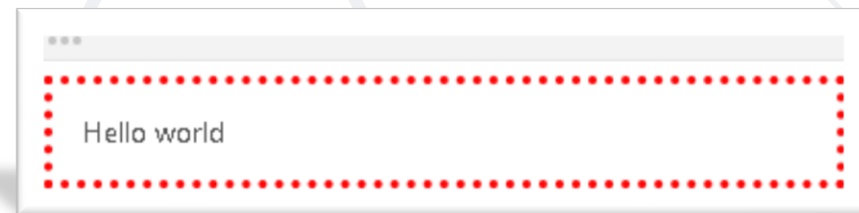
- Padding: shorthand property for padding-top, padding-right, padding-bottom, padding-left
- padding: 0;
- padding: 30px;



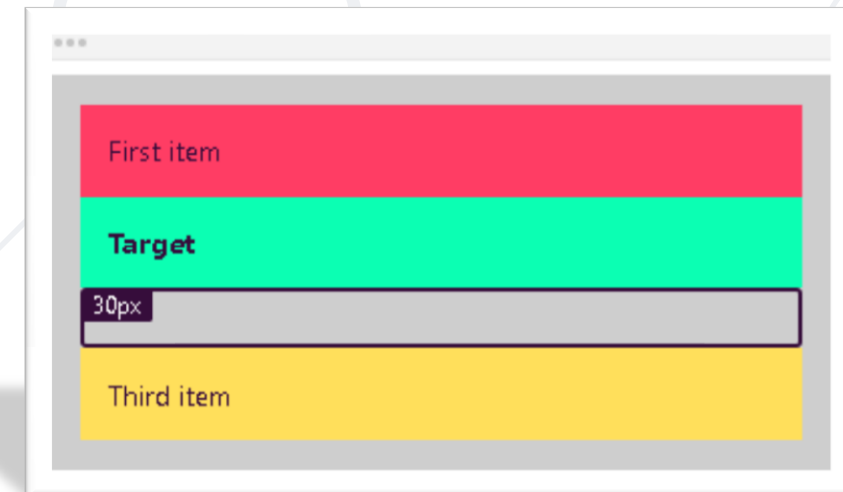
- `padding: 30px 60px;`
- `padding: 30px 60px 45px;`
- `padding: 30px 60px 45px 85px;`



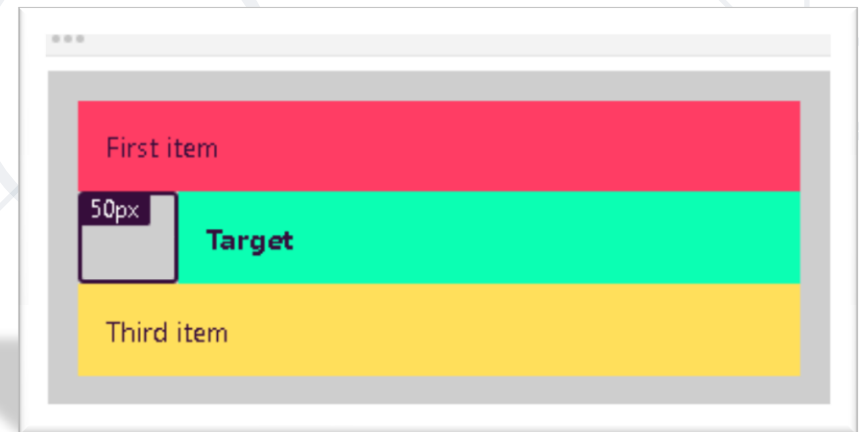
- Border: shorthand property for **border-width**, **border-style** and **border-color**
 - `border: 2px solid black;`
 - `border: 4px dotted red;`



- **Margin** – defines the space **outside** the element
- Margin-bottom:
 - `margin-bottom: 0;`
 - `margin-bottom: 30px;`



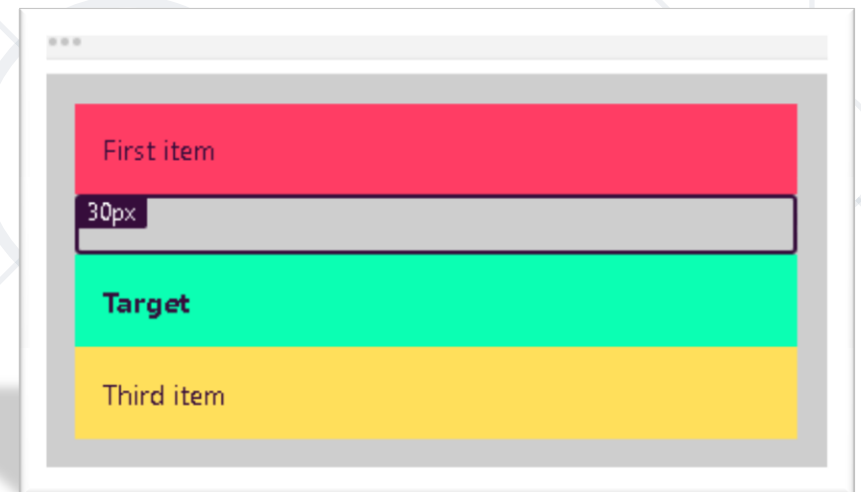
- Margin-left:
 - margin-left: 0;
 - margin-left: 50px;



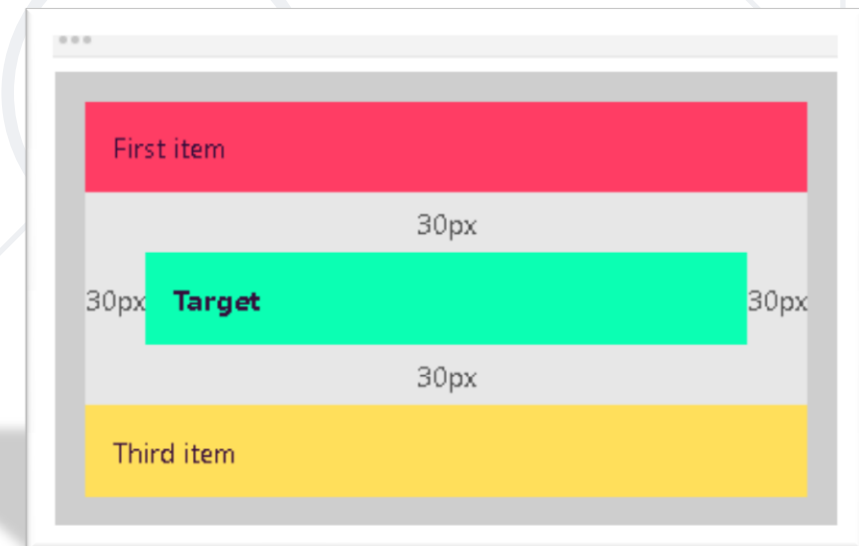
- Margin-right:
 - `margin-right: 0;`
 - `margin-right: 50px;`



- Margin-top:
 - `margin-top: 0;`
 - `margin-top: 50px;`



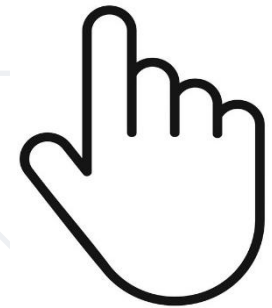
- Margin: shorthand property for margin-top, margin-right, margin-bottom, margin-left
 - `margin: 0;`
 - `margin: 30px;`



- **Cursor** – Sets the mouse cursor when hovering the element

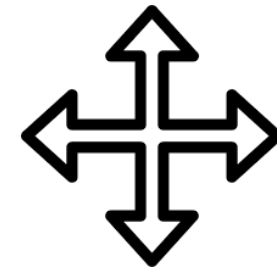
- Pointer

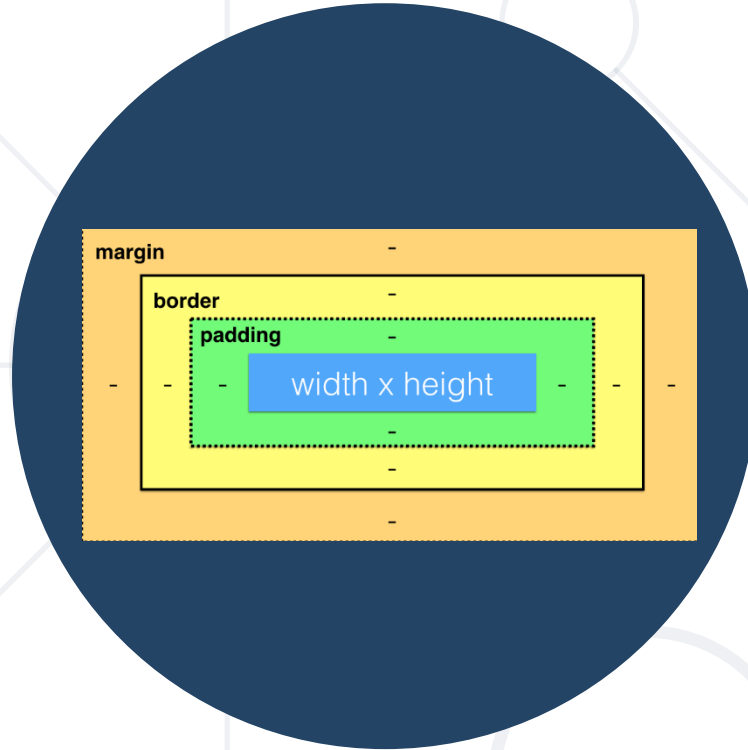
```
p {  
  cursor: pointer  
}
```



- Move

```
p {  
  cursor: move;  
}
```

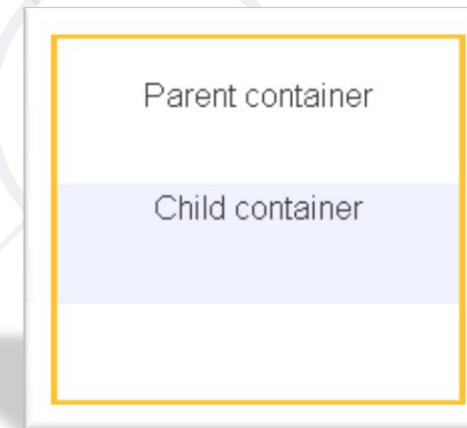




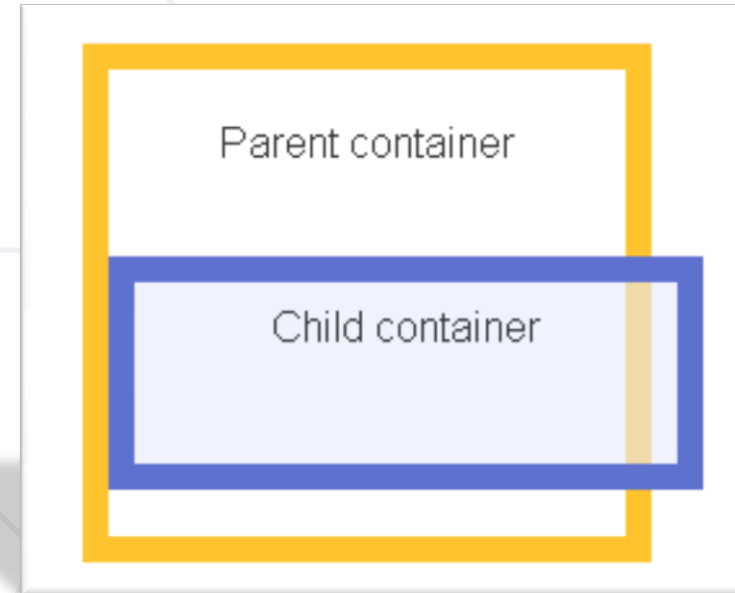
Box Sizing

- Box-sizing: sets how the total width and height of an element is calculated
 - **content-box** – initial and default value
 - The **width** and **height** properties include the content, but does **not include** the padding, border and margin

```
box-sizing: content-box;  
width: 200px;
```



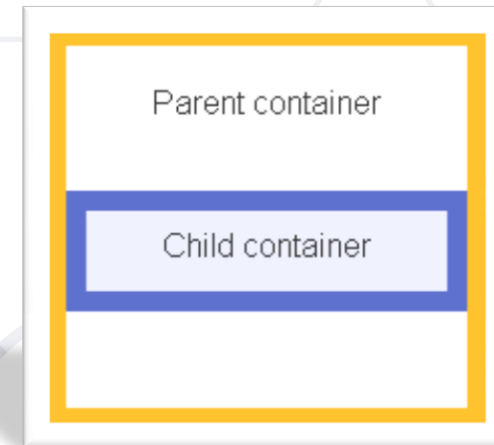
```
box-sizing: content-box;  
width: 200px;  
border: 10px solid #5B6DCD;  
padding: 5px;
```



- The full width is: $200\text{px} + 2 * 10\text{px} + 2 * 5\text{px} = 230\text{px}$

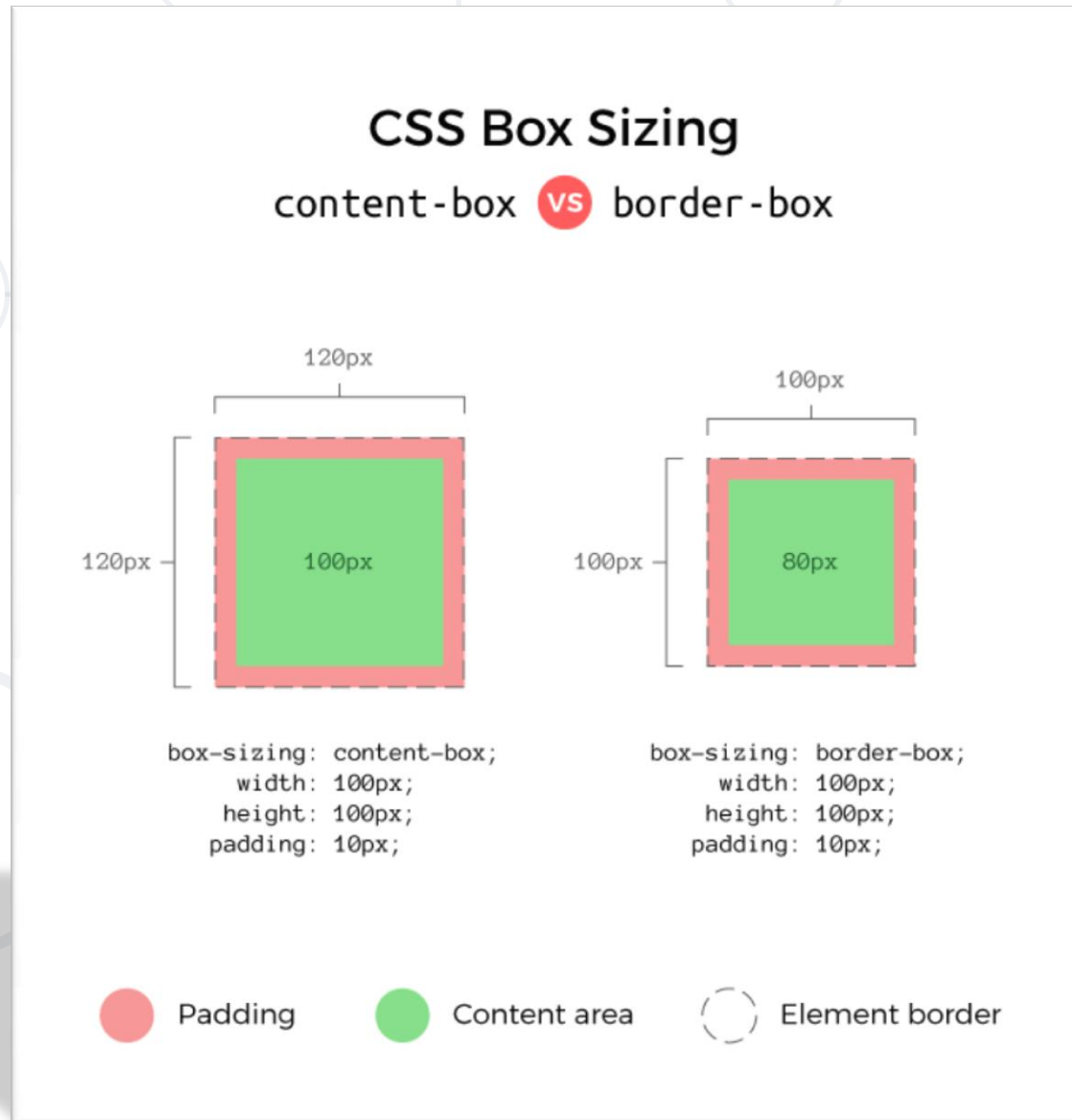
- **border-box** – the **width** and **height** of the element apply to all parts of the element: the **content**, the **padding** and the **borders**

```
box-sizing: border-box;  
width: 200px;  
border: 10px solid #5B6DCD;  
padding: 5px;
```



- The full width is **200px**
- The content width is equal to: $200\text{px} - 2 * 10\text{px} - 2 * 5\text{px} = 170\text{px}$

Content-box vs Border-box



- The box-sizing **Reset** takes care of the box-sizing of every element by setting it to border-box using universal CSS selector
- Save your **time** and don't write the same thing **again-and-again**
- Set the "universal box-sizing" with inheritance:

```
html {  
  box-sizing: border-box;  
}  
  
*, *:before, *:after {  
  box-sizing: inherit;  
}
```

- What is Box Model?
- Width and Height to the elements
- What are the padding, border and margin?
- What is box-sizing?
- How to reset Box-sizing?



SoftUni Diamond Partners



XSsoftware



SBTech
we know sports



telenor



SoftwareGroup
doing it right

NETPEAK



SmartIT



Postbank

Решения за твоето утре

**SUPER
HOSTING
.BG**

INDEAVR

Serving the high achievers



INFRAGISTICS®



STEMO®
Computer Systems & Software



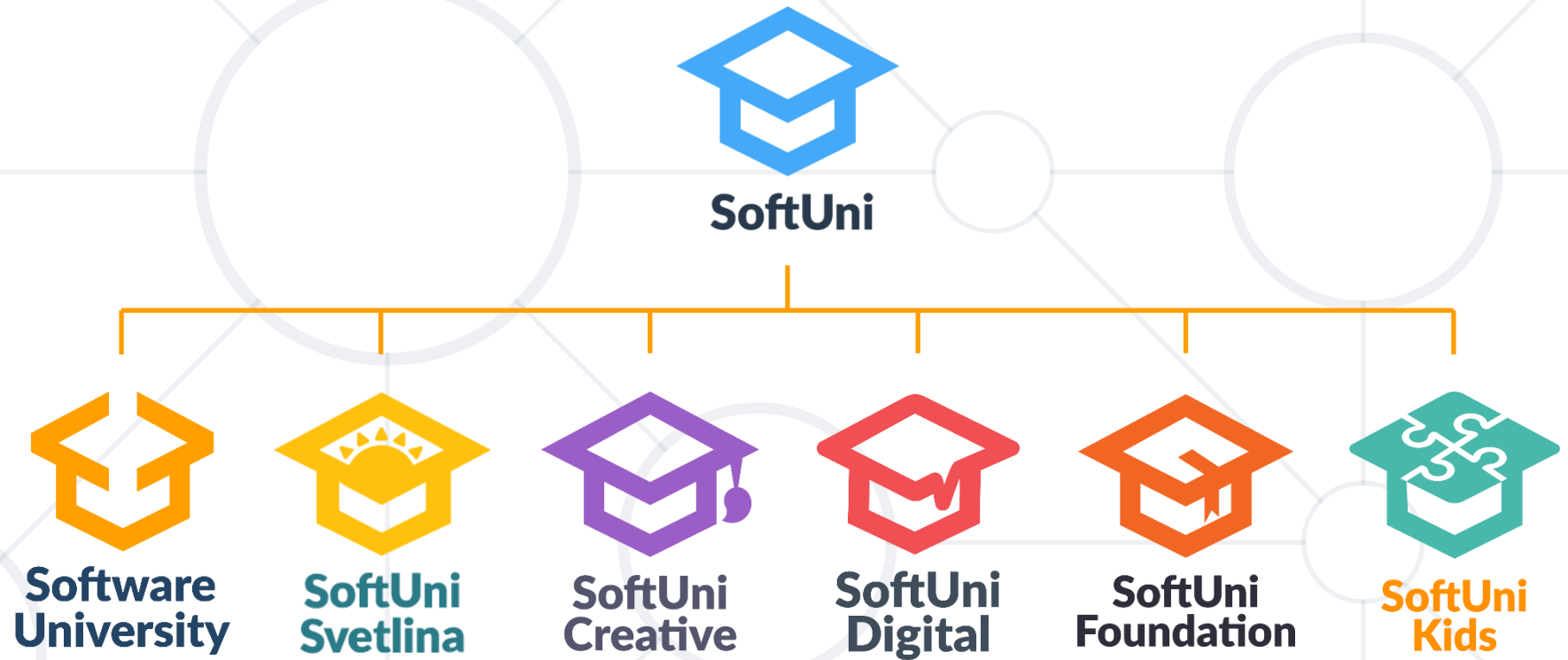
SoftUni Organizational Partners



OneBit
SOFTWARE



Questions?



- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – <https://softuni.org>
- © Software University – <https://softuni.bg>



- Software University – High-Quality Education, Profession and Job for Software Developers

- softuni.bg, softuni.org

- Software University Foundation

- softuni.foundation

- Software University @ Facebook

- facebook.com/SoftwareUniversity

- Software University Forums

- forum.softuni.bg

